⑲日本国特許庁（JP）　⑪特許出願公開

⑫公開特許公報（A）　昭61－107434

⑤Int.Cl.⁴	識別記号	庁内整理番号	㊸公開 昭和61年(1986)5月26日
G 06 F 9/38 12/08		C－7361－5B 8219－5B	

審査請求 未請求 発明の数 1 （全7頁）

㊹発明の名称　データ処理装置

㉑特　願　昭59－227773
㉒出　願　昭59(1984)10月31日

㉓発明者　新谷　洋一　国分寺市東恋ケ窪1丁目280番地　株式会社日立製作所中央研究所内
㉓発明者　庄内　亨　国分寺市東恋ケ窪1丁目280番地　株式会社日立製作所中央研究所内
㉑出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
㉑出願人　日立マイクロコンピュータエンジニアリング株式会社　小平市上水本町1479番地
㉔代理人　弁理士　高橋　明夫　外1名

最終頁に続く

明　細　書

発明の名称　データ処理装置

特許請求の範囲

1．プログラム及びデータを格納する主記憶もしくは主記憶とその写しを保持するキャッシュ記憶を有し、プログラムを構成する複数の命令を同時に処理し、概念的に先行する命令の処理結果を使用するもしくは処理結果に依存する命令の処理を該先行命令の処理結果を待たずして予測により開始するデータ処理装置において、演算結果を主記憶に格納するストア命令を予測して処理する場合には該予測の正否が判定されるまで該ストア命令が予測状態であることを表示する手段と、該ストア命令の演算結果及びそのストアアドレスを予測の正否が判定されるまで保持しておくストアデータ保持手段と、予測が正しかった場合は上記予測状態表示手段による表示を非予測状態に変更すると共に上記ストアデータ保持手段により保持されている内容を用いて演算結果の主記憶あるいはキャッシュ記憶への書き込みを行う制御手段と、予測が誤っていた場合は上記ストアデータ保持手段により保持されている予測の誤ったストア命令のデータを無効化する制御手段とを有することを特徴とするデータ処理装置。

2．上記ストア命令の予測は、先行する命令の演算結果の予測であることを特徴とする第1項記載のデータ処理装置。

3．上記ストア命令予測は、分岐命令の分岐判定結果の予測であることを特徴とする第1項記載のデータ処理装置。

4．上記ストア命令の予測は、分岐命令の分岐先命令列データの予測であることを特徴とする第1項記載のデータ処理装置。

発明の詳細な説明

〔発明の利用分野〕

本発明はディジタルコンピュータに係り、特に概念的に先行する命令の実行が終了しないうちにその結果等を予測しながら後続の命令を並列に実行することで高速化を図るデータ処理装置におけ

るストア処理方式に関する。

〔発明の背景〕

従来より、汎用大型ディジタルコンピュータにおいては、高速化のためにパイプライン方式や並列処理方式のような、複数の命令を同時に処理するのが一般的となつている。この例としては、日経エレクトロニクス「汎用コンピュータ」p.251～263 "IBM3033プロセッサの内部設計とパフォーマンズ" で挙げられているIBM3033, "An Efficient Algorithm for Exploiting Multiple Arithmetic Units" ,IBM Journal Jan. 1967 で挙げられているIBM360／91がある。

3033では高速化のために、分岐命令の処理においては、分岐判定が下る前に後続の命令の処理を開始するために、分岐が成立か不成立かの予測を立て、分岐判定が下るまでは後続命令を予測状態のまま処理する方式をとつている。

一方、360／91では高速化のために独立に命令の演算を行える演算器を複数設け、入力オペランドが揃い次第、概念的に後の命令であつても直ちに演算を始めるという方式をとつている。

さてより一層の高速化を図るためには、これら予測処理方式と並列演算方式の両方を採用することが望ましいが、この場合次のような問題が生ずる。すなわち、予測状態でストア命令の処理を行い結果を主記憶に書き込んだ後で実は予測が誤つていたことが判明する、というケースが起るとすると、2つの理由で好ましくない。第1の理由は、結局該ストア命令は実行してはいけなかつたわけであるから、主記憶の内容をその書き込みの前の状態に復元する必要が生ずるため、このための制御論理を設ける必要が生じ、またこの復元のための余分な時間ロスが生じるため高速化を妨げる可能性があることである。また第2の理由は、主記憶を複数のCPU（中央処理装置）やチヤネルが共有する場合、上記のストア命令が誤つて書き込んだ結果を、該CPUが主記憶の復元を完了する前に他のCPUやチヤネルが読み出してしまう可能性があるが、これは多くの場合許されないことである。

〔発明の目的〕

本発明の目的は予測処理方式と並列演算方式を共に採用しているデータ処理装置において上記従来技術の問題点のない、すなわち、予測中のストア命令による誤つた主記憶書き込みによる復元動作がなく、また誤つた主記憶の内容を他CPUやチヤネルから読み出されることのない、高速な命令処理方式を提供することにある。

〔発明の概要〕

本発明においては、ストアすべきデータ、アドレス情報を保持するストアバツフアを1つ以上設け、各ストアバツフアにストア命令の処理が予測中か否かの状態フラグを付与する。ストア命令の処理にあたつては、演算ステージまでは予測中か否かに関わらず、並列演算方式にのつとつて処理を続けるが、主記憶書き込みステージにおいては予測状態が解消するまでは主記憶への書き込みを行わず、ストアバツフアにデータ等を保持するのみにしておく。そして予測状態が解消した時点で、予測が誤つていればストアバツフア内のストア命令をキヤンセルし、予測が正しければ主記憶に書き込みを行う。

〔発明の実施例〕

以下本発明の実施例を説明する。説明の都合上IBMのシステム370アーキテクチヤに基づくマシンを前提とする。

第1図は典型的な命令フオーマツトを示している。第1図（a）は、浮動小数点加算命令ADや、ストア命令STのフオーマツトであり、OP部は演算の内容を、R1部は命令の第1オペランドが格納されるレジスタ番号を、X2部，B2部，D2部は、第2オペランドアドレスを作成するためのインデクスレジスタ番号，ベースレジスタ番号，デイスプレイスメントをそれぞれ示す。第1図（b）は、Branch on Condition命令（BCと略記）のフオーマツトであり、OP部はBC命令であることを、M1部は分岐成立とすべき条件コードの値を示すマスクを、X2，B2，D2部は分岐先命令アドレスを作成するためのインデクスレジスタ番号，ベースレジスタ番号，デイスプレイス

メントをそれぞれ示す。

第２図は本発明を用いたデータ処理装置において、高速処理が可能な命令列の主記憶上の配置である。第１の命令はＡＤ命令であり、演算結果に基づいて条件コードが設定される。第２のＢＣ命令はこの条件コードの値によつて分岐成立／不成立を判定する。第３のＳＴ命令は、該ＢＣ命令が分岐不成立であつた場合実行される。このＳＴ命令は、該ＢＣ命令の分岐判定により実行されるか否かが定まる点を除けば、該ＢＣ，ＡＤ命令とデータの依存関係はなく、またオペランドレジスタの値は十分前に確定しているものとする。

第３図は、本発明を適用したデータ処理装置の全体構成の概略図である。１及び２はＣＰＵ，３はプログラム及びデータを格納する主記憶とその制御を行う主記憶装置（以下ＭＳと略記する）、４はチヤネルであり、入出力装置５とＨＳとの間のデータ転送を制御する。ＣＰＵ１，２，チヤネル４は、いずれもＨＳ３と接続されており、それぞれ主記憶に対して読み出しと書き込みを行う。

ＣＰＵ－１の内部は命令制御ユニット（以下ＩＵと略す）６，演算ユニット（以下ＥＵと略す）７，記憶制御ユニット（以下ＳＣＵと略す）８から成る。

ＩＵ６はＳＣＵ８に対して命令読み出し要求を発行しＳＣＵ８経由で読み出された命令を解読しオペランド読み出し要求を再びＳＣＵ８に発行する。読み出されたオペランドはＳＣＵ８及びＩＵ６経由にてＥＵ７に送出される。ＩＵ６によつてオペランドと伴に命令の解読情報がＥＵ７に渡されると、ＥＵ７では複数個設けられている演算器Ｅ０，Ｅ１，…のうち空いているものにこれをセツトアツプし演算を行う。命令がストア命令である場合は、演算器はＳＣＵ８に対してストア要求を発行し、ストアデータ，アドレス情報を送出する。ＳＣＵ８はＩＵ６からの命令読み出し、オペランド読み出しＥＵ７からのストア要求を受け付け、必要ならアドレス変換を行い、ＨＳ３に対してこれらの主記憶参照を行う。ＳＣＵ８には通常の制御回路の他に本発明によるストアバツファ９とストアバツファ制御回路１０が設けられている。

第４図は、上記ＣＰＵ１がＭＳ３上に置かれた第２図の命令列を処理した時の概略のタイムチヤートである。横軸はマシンサイクルを単位とした時間を表わし、縦軸は命令処理を行う各論理ユニット／論理回路である。本タイムチヤートでは、各論理ユニット／論理回路において各命令の処理が行われている時間帯を、該命令のニーモニックを矩形で囲むことにより表している。またＩＵ６にてＡＤ命令の解読の始まる時刻からＳＴ命令によるＭＳ３への書き込みの終了時刻までを、順にＣ１，Ｃ２，…，Ｃｎと呼ぶことにする。

Ｃ１にてＡＤ命令の解読、オペランド読み出しが行われ、Ｃ２からＣ５の間、演算器Ｅ０にて演算が行われる。本タイムチヤートでは示していないがＣ６にて結果を浮動小数点レジスタにまた条件コードをＰＳＷに書き込む。またＢＣ命令はＣ２にて解読が行われ、Ｃ３にて演算器Ｅ１にセットアップされるが、先行するＡＤ命令の処理が完了していないため、ＡＤ命令が設定する条件コードを用いて行うべき分岐判定ができず、従ってＣｎまで保留される。Ｃ４にてＡＤ命令の条件コードが設定されるので同サイクルにおいてＢＣ命令の分岐判定が行われる。本例においては分岐不成立であつたものとする。一方ＳＴ命令はＣ３において解読を行う。本来Ｃ４ではＢＣ命令の分岐判定が下つていないため、ＳＴ命令を実行すべきか否か決定できないが、ＢＣ命令は分岐不成立になるという予測のもとにＳＴ命令の処理を開始する。ＳＴ命令はＣ４にて演算器Ｅ１にセットアップされ、Ｃ５にてストアデータやアドレス情報がストアバツファに一時的に格納される。この時このＳＴ命令が予測状態であることを示す予測フラグをセットする。一方Ｃ６にてＢＣ命令の分岐判定が下るが、本例では上述のように分岐不成立であるため、予測が正しかつたことが判明する。ゆえに、ストアバツファ内に格納されていた該ＳＴ命令の予測フラグをリセットしさらにストアデータを、アドレス情報に従ってＣｎにて主記憶に書き込む。仮に上記ＢＣ命令の分岐判定により、予測が誤つて

いたとすると、この場合は、ストアバッファ内の該ＳＴ命令をキャンセルする。このようにすることで、演算器Ｅ,はＣ,にて空き状態にすることができ、後続命令の演算に使用可能となり、他の演算器が使用である時には命令処理の高速化に役立つ。

次に第５図，第６図を用いてストアバッファ及びストアバッファ制御回路の構成を説明する。

まず第５図はストアバッファの構成を示す。５０１～５０２はｍ＋１個のストアデータレジスタ及びストアアドレスレジスタである。演算ユニットにてＳＴ命令が実行されると、演算ユニットより信号線５０３を介してストア要求信号ＲＥＱが発行され、同時に信号線５０４，５０５を介して演算ユニットから送られてくるストアデータ及びストアアドレスを５０１～５０２のいずれかのレジスタに格納する。この場合何番のレジスタに格納するかは、ストアバッファ制御回路１０より信号線５０６を介して送られてくる入力ポインタＩＰで示される。５０７はセレクタであり、ストアバッファ出力制御回路より信号線５０８を介して送られてくる出力ポインタＯＰで示される番号のストアデータレジスタ及びストアアドレスレジスタの内容を選択し、これを信号線５０９を介して主記憶装置に送出する。

第６図はストアバッファ制御回路１０の構成を示す。６０１はストアバッファ状態制御回路であり、上述した５０１～５０２のｍ＋１個のレジスタ対応して、ストア要求の有無を示す要求フラグと、ストア要求が予測状態か否かを示す予測フラグ６０２～６０３とを有する。すなわち、６０２は、ストアバッファ内のｎ番のレジスタに対応してストア要求が有ることを示す要求フラグＲＥＱ。及び、該ストア要求が予測状態で処理しているＳＴ命令によるものであることを示す予測フラグＰ。である。また６０３はｍ番のレジスタに対応する要求フラグＲＥＱ。と予測フラグＰ。である。

６０４はストアバッファ入力制御回路である。６０４は６０１から信号線６０５を介して

ＲＥＱ。～ＲＥＱ。を受けとり、このうちオフであるものの中から１つを選択しその番号をストアバッファの入力ポインタＩＰとして信号線５０６に出力する。信号線５０６はストアバッファ９及びストアバッファ状態制御回路６０１に送出される。ＥＵ７（第３図）から信号線５０３を介してストアバッファ状態制御回路にＲＥＱ信号が送出されると、入力ポインタＩＰで示される番号の要求フラグを１にセットし、同時に信号線６０６を介してＥＵ７から送出される予測状態信号Ｐの値を入力ポインタＩＰで示される予測フラグの値としてセットする。

６０７はストアバッファ出力制御回路である。６０７は６０１から信号線６０５を介してＲＥＱ。～ＲＥＱ。を、また信号線６０８を介してＰ。～Ｐ。を受けとり、このうち予測フラグがオフでかつ要求フラグがオンであるレジスタの番号をストアバッファの出力ポインタＯＰとして信号線５０８に出力する。信号線５０８は、ストアバッファ９及びストアバッファ状態制御回路６０１に送出される。また実際に予測フラグがオフでかつ要求フラグがオンであるレジスタが存在した場合は、６０７はＭＳ３（第３図）に対するストア要求信号NREQを発行する。NREQは信号線６０９を介して、３及び６０１に送出される。６０７がNREQを発行した時、６０１はＯＰで示される番号の要求フラグ及び予測フラグを０にリセットする。

一方、ストアバッファ状態制御回路６０１は、ＥＵ７から信号線６１０を介して送られてくる分岐判定信号ＢＪを受けとると、もし分岐予測が正しければ、すなわち本実施例においては、ＢＪが分岐不成立を示していたならば、Ｐ。～Ｐ。をすべて０にリセットする。こうすることにより、今まで要求フラグはオンであったにも係らず予測フラグがオンであったがためにNREQを発行できないでいたストア要求は、予測フラグがオフとなるためNREQを発行することができ、ＭＳに書き込みが行われることとなる。またもしＢＪ信号を受け取った時に分岐予測が誤っていたとすれば、ＲＥＱ。～ＲＥＱ。のうち、対応する予測フラグがオンであ

るものを全て'0'にリセットする。同時にP。～P。も全て'0'にリセットする。こうすることにより、ストアバッファ内の、実は実行してはいけなかつたストア要求はキャンセルされたことになる。

演算ユニットから記憶制御ユニットSCL8に信号線503，504，505，610を介して送出される信号REQ，ストアデータ，ストアアドレス，BJはいずれも従来技術で容易に作成することができるのでその説明は省略する。また信号線606上の信号Pは例えば次のようにして作成できる。すなわち、命令制御ユニット6からEU－7への命令セットアップは命令の概念的実行順序通りだとした場合、EU7へ分岐命令がセットアップされたと同時に'1'にセットされ、分岐判定が行われたと同時に'0'にセットされるフリップフロップの値をPとすればよい。

またSCU8からMS3に信号線609，509を介して送出される信号MREQ，ストアデータ，ストアアドレスに対するMS－3の構成とこれらを受けとつた時の、動作は従来技術と同様であるので説明を省略する。

なお上記実施例における予測としては、分岐命令の分岐判定だけではなく、次に述べるような各種のものが考えられ、いずれに対しても本発明を適用可能である。すなわち、特公昭54-9456にあるように、分岐命令における分岐先命令データを予測するもの、特願昭58-237778にあるように、レジスタコンフリクトを生じる2命令の処理において先行命令の演算結果を予測するものがある。

〔発明の効果〕

本発明によれば、予測処理方式と並列演算方式を共に採用しているデータ処理装置において、予測中であつてもストア命令の演算処理を完了し、該ストア命令の演算処理を行つた演算器がさらに別の命令の演算処理に使用可能となるため、高速化に効果がある。しかも本発明によれば予測中の命令によつて誤つて主記憶に書き込みが行われることがないため、復元動作が必要なく、これによる処理速度低下も生じない。さらに誤つて主記憶に書き込みが行われることがないため、他CPUやチャンネルからこの予測の誤つたストア命令の結果が見えてしまうことも妨げるため、この点についてIBMシステム370アーキテクチャ等の仕様を守ることも可能である。

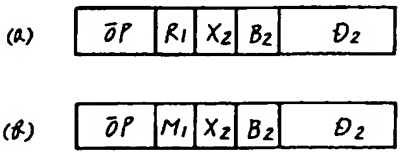
図面の簡単な説明

第1図は典型的な命令フォーマットを示す図、第2図は高速化される命令列の例を示す図、第3図はデータ処理装置の全体構成例を示す図、第4図は第2図の命令列を処理した時のタイムチャート、第5図はストアバッファの構成図、第6図はストアバッファ制御回路図である。

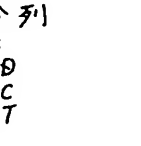
3…主記憶装置、7…演算ユニット、8…記憶制御ユニット、9…ストアバッファ、10…ストアバッファ制御回路、601…ストアバッファ状態制御回路、604…ストアバッファ入力制御回路、607…ストアバッファ出力制御回路。

代理人　弁理士　高橋明夫

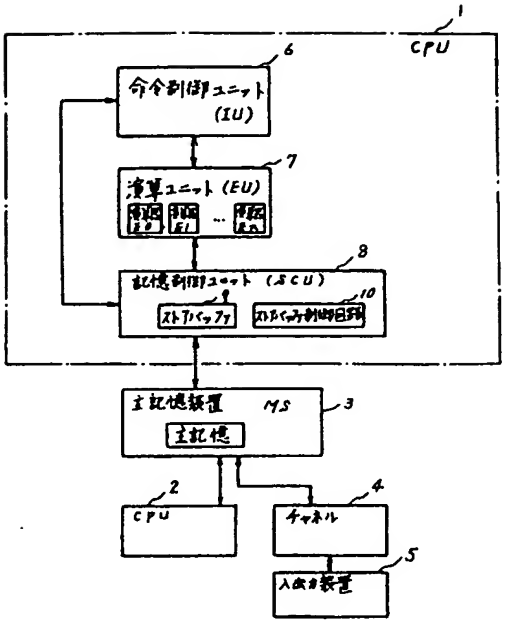
第1図



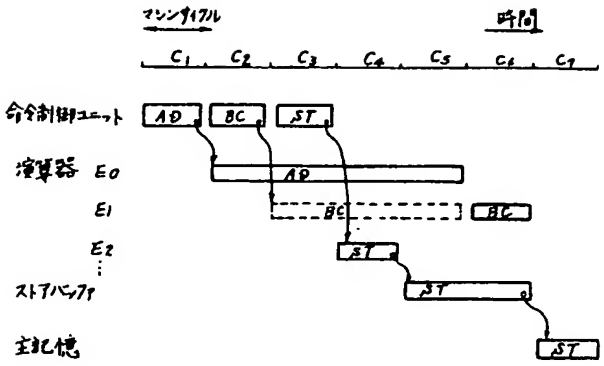
第2図



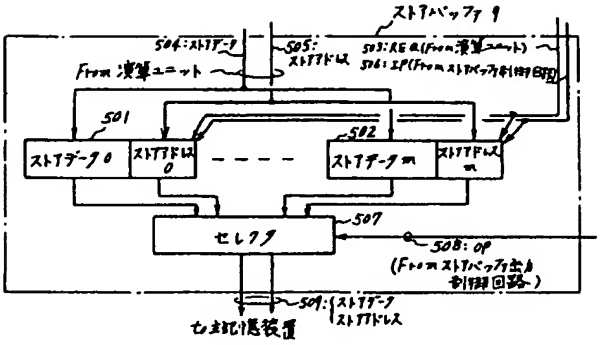
第 3 図



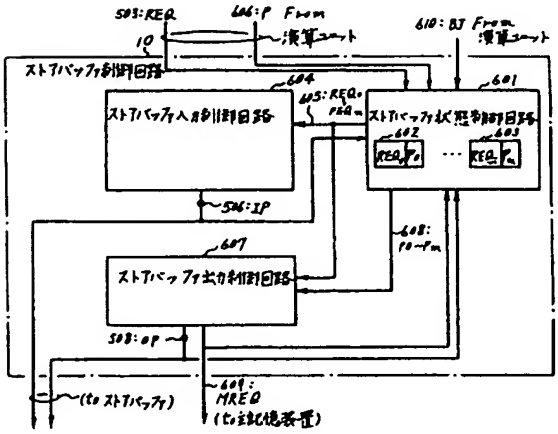
第 4 図



第 5 図



第 6 図



第1頁の続き
⑫発明者　釜田　栄樹　国分寺市東恋ケ窪1丁目280番地　株式会社日立製作所中央研究所内
⑫発明者　武内　茂雄　小平市上水本町1479番地　日立マイクロコンピュータエンジニアリング株式会社内

(19) Japan Patent Office (JP)

(11) Publication of Patent Application

Sho 61 – 107434

(12) Publication of Japanese Laid-Open Patent Application (A)

(43) Date of Publication: May 26, 1986

(51) Int. Cl[4]	Class. Symbol	Internal No.
G 06 F 9/38		C – 7361 – 5B
12/08		8219 – 5B

Request for Examination: Not requested

Number of Inventions: 1

(Total of 7 pages in the original)

(54) Title of the Invention: DATA PROCESSOR

(21) Japanese Patent Application: Sho 59 – 227773

(22) Filing date: October 31, 1984

(72) Inventor: Yoichi SHINTANI

Hitachi, Ltd., Center Research Institute

280, Higashikoigakubo 1-chome, Kokubunji-shi

(72) Inventor: Tohru SHONAI

Hitachi, Ltd., Center Research Institute

280, Higashikoigakubo 1-chome, Kokubunji-shi

(71) Applicant: Hitachi, Ltd.

6, Kandasurugadai 4-chome, Chiyoda-ku, Tokyo

(71) Applicant: Hitachi Microcomputer Engineering

(74) Agent: Patent agent: Akio TAKAHASHI and one other

SPECIFICATION

TITLE OF THE INVENTION:

DATA PROCESSOR

SCOPE OF PATENT CLAIMS:

Claim 1

A data processor, wherein, in a data processor that has a cache memory, which maintains a main memory where programs and data are stored, or which maintains a main memory and its copy; and that simultaneously processes multiple instructions comprising the programs; and that starts processing an instruction using the processing results of a conceptually precedent instruction, or depending upon the processing results, without waiting for the processing results of the precedent instruction, it is provided with:

a means where, in the case of processing by predicting a storage instruction, whose operational results are stored in the main memory, it indicates that the storage instruction is in the prediction state until a determination can be made as to whether or not the prediction is correct; a storage data maintaining means that maintains the result of an operation of the storage instruction and its storage address until whether or not the prediction is correct is determined; and

a control means that changes the indication by the prediction state indicating means into a non-prediction state; concurrently, that writes the result of an operation using the contents maintained by the storage data maintaining means, into the main memory or the cache memory in the case that the prediction is correct; and

another control means that invalidates the data of the storage instruction maintained by the storage data maintaining means according to the incorrect prediction, in the case that the prediction is incorrect.

Claim 2

The data processor according to Claim 1, wherein, the prediction of the storage instruction is the prediction of the results of the operation of the precedent instruction.

Claim 3

The data processor according to Claim 1, wherein, the prediction of the storage instruction is a prediction of a branch decision result of a branch instruction.

Claim 4

The data processor according to Claim 1, wherein, the prediction of the storage instruction is a prediction of a branch target instruction array data of a branch instruction.

Detailed Description of the Invention:

[Application Field of the Invention]

The present invention relates to a digital computer, and particularly relates to a storage processing method in a data processor for the purpose of increasing the processing speed, where before the completion of the execution of a precedent instruction, the results are generally conceptually predicted; and simultaneously, successive instructions are executed in parallel.

[Background of the Invention]

Conventionally, in a general-purpose large-sized digital computer, for the purpose of increasing the processing speed, it has become common to simultaneously process

multiple instructions, such as a pipeline method or a parallel processing method. As this example, there are the IBM 3033, which is mentioned in the "Internal Design and Performance of IBM 3033 Processor", "General-purpose Computer", Nikkei Electronics, p. 251 – 263, and the IBM 360/91, which is mentioned in "An Efficient Algorithm for Exploiting Multiple Arithmetic Units", IBM Journal, Jan. 1967. In the [IBM] 3033, for the purpose of increasing the processing speed, in the processing of a branch instruction, it has adopted a method where in order to start processing of successive instructions before a decision about a branch is made, whether or not the branch is realized is predicted, and the successive instructions are processed in the prediction state until the branch decision is made.

In the meantime, in the [IBM] 360/91, a method is adopted where multiple computing elements that can independently perform instruction operations are established for the purpose of increasing the processing speed, and as soon as input operands are gathered, even if it is a conceptually successive instruction, the operation immediately is started.

In order to accomplish an even higher speed, it is desirable to adopt both the prediction processing method and the parallel operation method. However, in this case, the following

problems occur. In other words, in a case where the processing of a storage instruction is performed in the prediction state, and after the results have been written into main memory, if it turns out that the prediction is actually wrong, it is not preferable for two reasons. The 1st reason is because the storage instruction should not have been executed, so it becomes necessary to restore the contents of the main memory to the state existent before writing was performed. Therefore, it is necessary to establish a control logic, and there is a possibility where it may prevent increasing the processing speed because an extra time loss is generated due to the restoration. The 2nd reason is because, in the case that multiple CPUs (central processing units) and channels share the main memory, there is a possibility that results which have been improperly written by the above-mentioned storage instruction, may be read out by another CPU or channel before the CPU completes the restoration of the main memory. However, this is not allowed in many cases.

[Objective of the Invention]

The objective of the present invention is to provide a high-speed instruction processing method where in a data processor, where both the prediction processing method and the parallel operation method have been adopted, there is no problem such as that of the prior

art. In other words, any restoration operation due to improper writing into the main memory by a storage instruction during the prediction state is not required, and any wrong content in the main memory will not be read out by other CPUs and channels.

[Summary of the Invention]

In the present invention, one or more storage buffers are established to maintain data and address information to be stored, and a state flag that indicates whether or not the processing of the storage instruction is in the prediction state is provided to each storage buffer. In processing the storage instruction, the processing is continued to the operation stage based upon the parallel operation method, regardless of whether or not it is in the prediction state. However, in the stage of writing into the main memory, writing into the main memory is not performed until the prediction state is resolved, and data generally are maintained in the storage buffer. At the point at which the prediction state is resolved, if the prediction is incorrect, the storage instruction within the storage buffer is cancelled, and if the prediction is correct, writing into the main memory is performed.

[Embodiments of the Invention]

An embodiment of the present invention is explained next. For the convenience of explanation, the machine [example] is based on the IBM system 370 architecture.

Fig. 1 shows typical instruction formats. Fig. 1 (a) is the format of a floating point add instruction AD or a storage instruction ST, and the OP part indicates the contents of the operation; R_1 indicates a register number where a 1st operand of the instruction is stored; X_2, B_2 and D_2 indicate an index register number, a pace register number and a displacement for the purpose of forming a 2nd operand address, respectively. Fig. 1 (b) is the format of a Branch-on-Condition instruction (hereafter abbreviated as BC), and OP indicates that it is a BC instruction. Part M_1 indicates a mask that indicates a value for a condition code to realize a branch; and X_2, B_2 and D_2 indicate an index register number, a pace register number and displacement for the purpose of forming the branch target address, respectively.

Fig. 2 is an arrangement in the main memory of an instruction array where high-speed processing is realizable in a data processor using the present invention. The 1st instruction

is an AD instruction, in relation to which a condition code is established based on the results of the operation. The 2nd instruction is a BC instruction for making a decision as to whether or not the branch is realized according to the value of the condition code. The 3rd instruction is an ST instruction executed in the case that the branch in the BC instruction is not realized. The ST instruction does not have any dependency with the BC and AD instructions regarding data, except for a point where whether or not the ST instruction is executed is determined according to the branch decision of the BC instruction, where the value for the operand register is ascertained sufficiently in advance.

Fig. 3 is an outline of the entire construction of the data processor where the present invention has been adopted. Symbols '1' and '2' are CPUs; '3' is a main memory where programs and data are stored and a main memory device where its control is performed (hereafter, abbreviated as 'MS'); '4' is a channel, which controls data transmission between an input/output device 5 and the HS [3][sic]. The CPUs 1 & 2 and the channel 4 are connected to the HS 3*[sic]*, respectively, and each of them writes and reads out to/from the main memory.

The interior of the CPU 1 is comprised of an instruction control unit (hereafter abbreviated as 'IU') 6, an operational unit (abbreviated as 'EU') 7 and a memory control unit (hereafter, abbreviated as 'SCU') 8.

The IU 6 issues an instruction read-out request to the SCU 8, decodes the read-out instruction via the SCU 8; and issues an operand read-out request to the SCU 8 again. The read-out operand is transmitted to the EU 7 via the SCU 8 and the IU 6. When the IU 6 passes the decoded information of the instruction along with the operand to the EU 7, it is set up in an available computing element among established multiple elements E0, E1, ..., and the operation is performed. In the case that the instruction is a storage instruction, the computing element issues a storage request to the SCU 8, and the storage data and the address information are transmitted. The SCU 8 reads out the instruction from the IU 6, reads out its operand, receives the storage instruction, and, the address conversion is performed if necessary, and then, these main memory references are performed to the HS 3[sic]. Other than a normal control circuit, a storage buffer 9 and a storage buffer control circuit 10 according to the present invention are established in the SCU 8.

Fig. 4 is an outlined time chart when the instruction array in Fig. 2, where the above-mentioned CPU 1 is placed on the MS 3, is processed. The horizontal axis indicates the time by unit of machine cycle, and the vertical axis indicates each logical unit/logic circuit. In the present chart, the time zone when each instruction is processed in each logical unit/logical circuit is expressed by surrounding a mnemonic of the instruction with a rectangular box. Further, in the IU 6, from the time for starting to decode the AD instruction to the time for completing to write into the MS 3 by the ST instruction is referred to as C_1, C_2, ..., $C_{[illegible]}$, in respective order.

Decoding the AD instruction and reading-out of the operand are performed during the C_1, and an operation is performed at the computing element E_0 during C_2 through C_5. Although not shown in the present time chart, the results are written into the floating point register and a condition code is written into the PSW during the $C_{[illegible]}$. Further, the BC instruction is decoded during C_2, and is set up to the computing element E_1 during C_3. However, processing the precedent AD instruction has not yet been completed, so the branch decision which should be performed using the condition code established by the AD instruction, cannot be made, and it is deferred until C_5. The condition code of the AD instruction is established during C_6, so the branch decision of the BC instruction is

performed during this cycle. In the present embodiment, it is presumed that branching is not realized. In the meantime, the ST instruction is decoded during C_3. Originally, the branch decision of the BC instruction is not made during C_3, so whether or not the ST instruction is executed cannot be determined. However, processing the ST instruction is started based upon the prediction where the branch of the BC instruction is not realized. The ST instruction is set up in the computing element E_2 during C_4, and the storage data and the address information are temporarily stored in the storage buffer during C_5. On this occasion, a prediction flag indicating the ST instruction to be in the prediction state is set. In the meantime, the branch decision of the BC instruction is made during C_6. As described above, branching is not in the present embodiment, so the prediction is correct. Therefore, the prediction flag of the ST instruction stored within the storage buffer, is reset. In addition, the storage data is written into the main memory during C_7 based on the address information. If the prediction is assumed to be incorrect according to the branch decision of the BC instruction, the ST instruction within the storage buffer is cancelled. The operation enables the computing element E1 to become vacant during C_5, making it usable for the operation of successive instructions. Therefore, when other computing elements are in use, it is helpful for increasing the speed of instruction processing.

The construction of the storage buffer and the storage buffer control circuit are explained hereafter, with reference to Fig. 5 and Fig. 6.

At first, Fig. 5 shows the construction of the storage buffer. Symbols '501' through '502' are (m+1) units of the storage data registers and the storage address registers, respectively. When the ST instruction is executed in an operational unit, a storage request signal REQ is issued from the operational unit via a signal line 503. Simultaneously, the storage data and storage address transmitted from the operational unit via signal lines 504 and 505, are stored in either of registers 501 through 502. In this case, information about in which number of register the storage data and the storage address are stored is indicated with an input pointer IP transmitted from the storage buffer control circuit 10 via signal line 506. The symbol '507' is a selector which selects the contents of the storage data register and the storage address register in the number indicated by an output pointer OP, which is transmitted from a storage buffer output control circuit via signal line 508, and then transmitted to the main memory device via signal line 509.

Fig. 6 shows the construction of the storage buffer control circuit 10. Symbol '601' is a storage buffer state control circuit, equipped with prediction flags 602 through 603 that

indicate whether the storage request is in the prediction state, by corresponding to the (m+1) units of registers 501 through 502. In other words, '602' is a request flag REQ, which indicates that there is a storage request corresponding to No. 0 register within the storage buffer, and is also a prediction flag P_0 which indicates that the storage request corresponds to the ST instruction being processed in the prediction state of the storage request. Further, symbol '603' is a request flag REQ_m and a prediction flag P_m corresponding to the No. *m* register.

Symbol '604' is a storage buffer input control circuit which receives the REQ_0 through the REQ_m from the [storage buffer state control circuit] 601 via a signal line 605. It selects one from these requests that are in the OFF state, and the number is transmitted to the signal line 506 as the input point IP of the storage buffer. The signal line 506 is sent to the storage buffer 9 and the storage buffer state control circuit 601. When an REQ signal is transmitted to the storage buffer state control circuit [601] from the EU 7 (Fig. 3) via the signal line 503, the request flag of the number indicated by the input point IP, is simultaneously set to '1', and the value for the prediction state signal P transmitted from the EU 7 via the signal line 606, is set as the value for the prediction flag indicated by the input point IP.

Symbol '607' is a storage buffer output storage circuit which receives the REQ_0 through REQ_m from the [storage buffer state control circuit] 601 via the signal line 605, and receives P_0 through P_m via the signal line 608, and register number(s), where the prediction flag is in the OFF state and the request flag is in the ON state are transmitted to the signal line 508 as the output point OP of the storage buffer. The signal line 508 is sent to the storage buffer 9 and the storage buffer state control circuit 601. Further, where a register for which the prediction flag is in the OFF state and the request flag is in the ON state, actually exists, the [storage buffer output control circuit] 607 issues a storage request signal MREQ to the MS 3 (Fig. 3). The [storage request signal] MREQ is transmitted to the [MS] 3 and the [storage buffer state control circuit] 601 via a signal line 609. When the [storage buffer output control unit] 607 issues the [storage request signal] MREQ, the [storage buffer state control circuit] 601 resets the request flag(s) and the prediction flag(s) of the number(s), which are indicated with the OP, to '0'.

In the meantime, when the storage buffer state control unit 601 receives a branch decision signal BJ transmitted from the EU 7 via a signal line 610, if the branch prediction is correct; in other words, in the present embodiment, if the BJ indicates that branching is not realized, P_0 through P_m are all reset to '0'. With this operation, storage request(s) where, because the

request flag is in the ON state but the prediction flag is also in the ON state, the MREQ cannot be issued, can be issued since the prediction flag becomes in the OFF state. Then, writing is performed into the MS [3]. Further, when the BJ signal is received, if the branch prediction is incorrect, requests where the correspondent prediction flags are in the ON state among REQ_0 through REQ_m are all reset to '0'. Simultaneously, P_0 through P_m are also all reset to '0'. With this operation, the storage request within the storage buffer, which should not have been actually executed, should be cancelled.

The signals REQ, the storage data, the storage addresses and the [branch decision signal] BJ transmitted from the operational unit to the memory control unit SCL 8 via the signal lines 503, 504, 505 and 610, can be easily formed using prior art technology, so the explanation is omitted. Further, the signal P on the signal line 606 is, for example, formed as follows. In other words, in the case that the instruction setup from the instruction control unit 6 to the EU 7 is the same as in the conceptual execution sequence of the instruction, a value for a flip-flop such that when a branch instruction is set up to the EU 7, and simultaneously, '1' is set, and, when the branch decision is made, simultaneously, '0' is set, can be considered as the signal P.

Further, the construction of the MS 3 relative to the signal MREQ, the storage data and the storage address transmitted from the SCU 8 to the MS 3 via the signal lines 609 and 509, and operation when these are received are similar to those in the prior art, so the explanation shall be omitted.

Furthermore, as a prediction in the above-mentioned embodiment, not only the branch decision of the branch instruction but also the various below-mentioned ones can be considered, to which the present invention is also applicable. In other words, one predicts a branch target instruction data in a branch instruction, as mentioned in Japanese Patent Publication Sho 54 – 9456, and there is another predicts the result of the operation of a precedent instruction in the processing of two instructions where a register conflict may occur, as it is mentioned in Japanese Patent Application Sho 58 – 237778.

[Efficacy of the Invention]

According to the present invention, in a data processor where both the prediction processing method and the parallel operation method are adopted, operational processing of the storage instruction is completed even if in the prediction state, and a computing element, which has performed the operational processing of the storage instruction,

becomes usable for the operational processing of another instruction. Hence, it is efficient for increasing the processing speed. Further, according to the present invention, writing into a main memory is never performed by an instruction in the prediction state, so a restoration operation is not required. Therefore, no processing speed reduction occurs. In addition, writing is never performed in error, so it prevents other CPUs or channels from reading out the results of storage instructions where its prediction is incorrect. Concerning this point, it is generally possible to keep the specifications, such as those of the IBM System 370 architecture..

BRIEF DESCRIPTION OF DRAWINGS:

Fig. 1 is a diagram that shows typical instruction formats;

Fig. 2 is a diagram that shows an example of an instruction array where the increase of the processing speed is realized;

Fig. 3 is a diagram that shows the entire construction example of the data processor;

Fig. 4 is a time chart when the instruction array in Fig. 2 is processed;

Fig. 5 is a block diagram of the storage buffer; and Fig. 6 is a storage buffer control circuit diagram.

3 ... main memory device, 7 ... operational unit, 8 ... memory control unit, 9 ... storage buffer, 10 ... storage buffer control circuit, 601 ... storage buffer state control circuit, 604 ... storage buffer input control circuit, 607 ... storage buffer output control circuit.

Agent: Patent applicant: Akio TAKAHASHI [seal]

FIG. 1

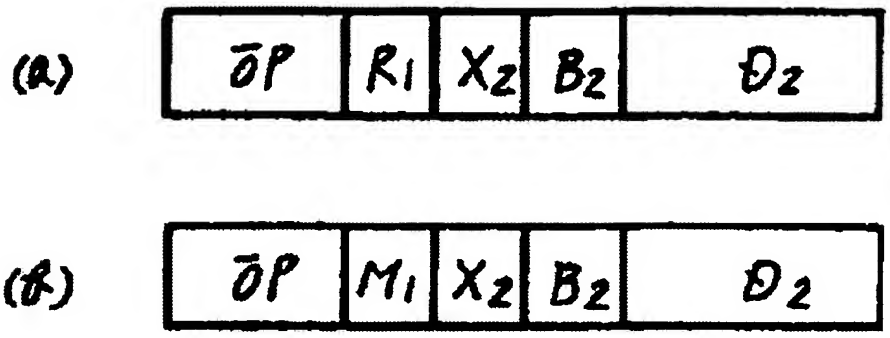


FIG. 2

FIG. 3

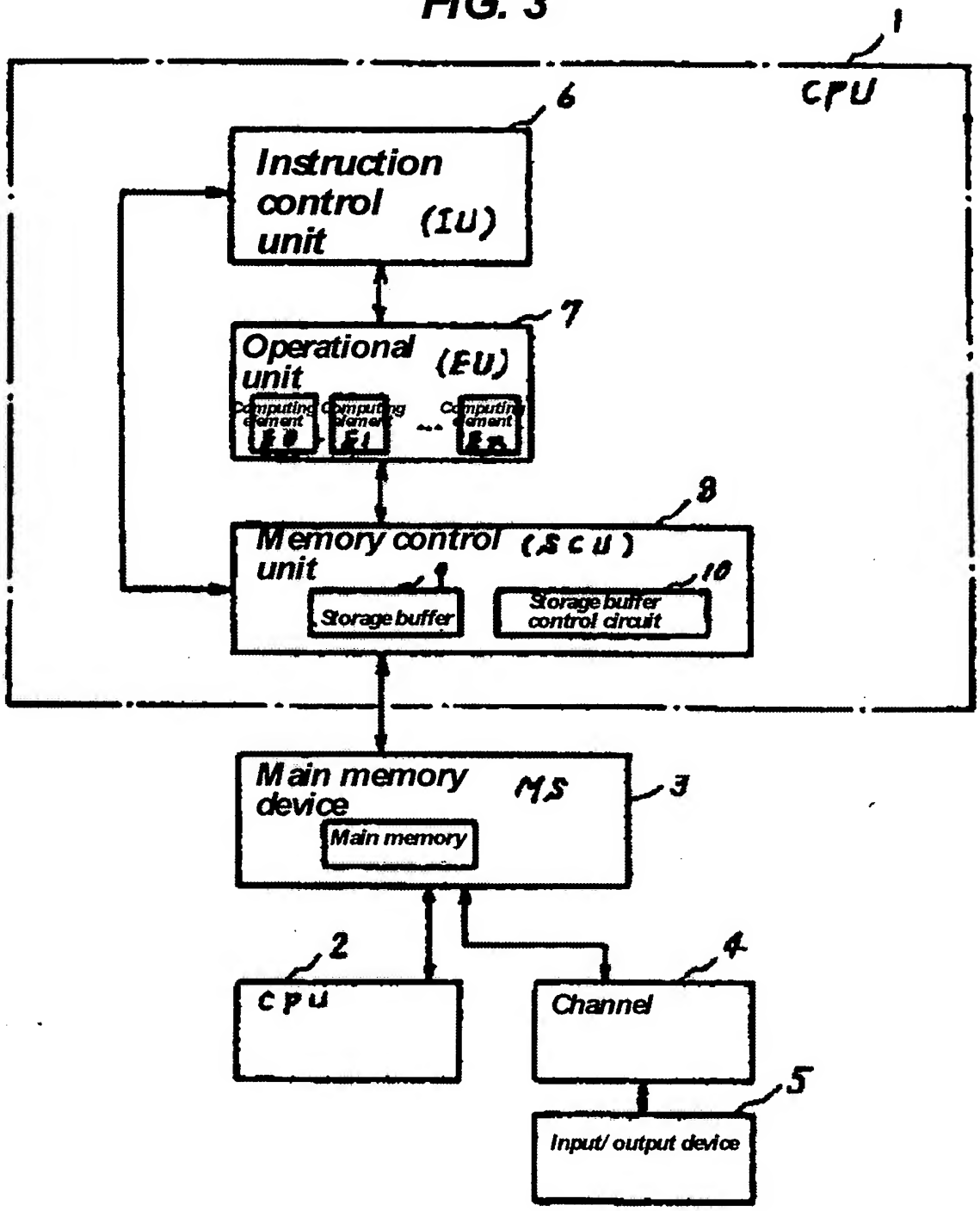

1
CPU
6
Instruction control unit (IU)
7
Operational unit (EU)
Computing element
Computing element
Computing element
8
Memory control unit (SCU)
Storage buffer
10
Storage buffer control circuit
Main memory device MS
3
Main memory
2
CPU
4
Channel
5
Input/ output device

FIG. 4

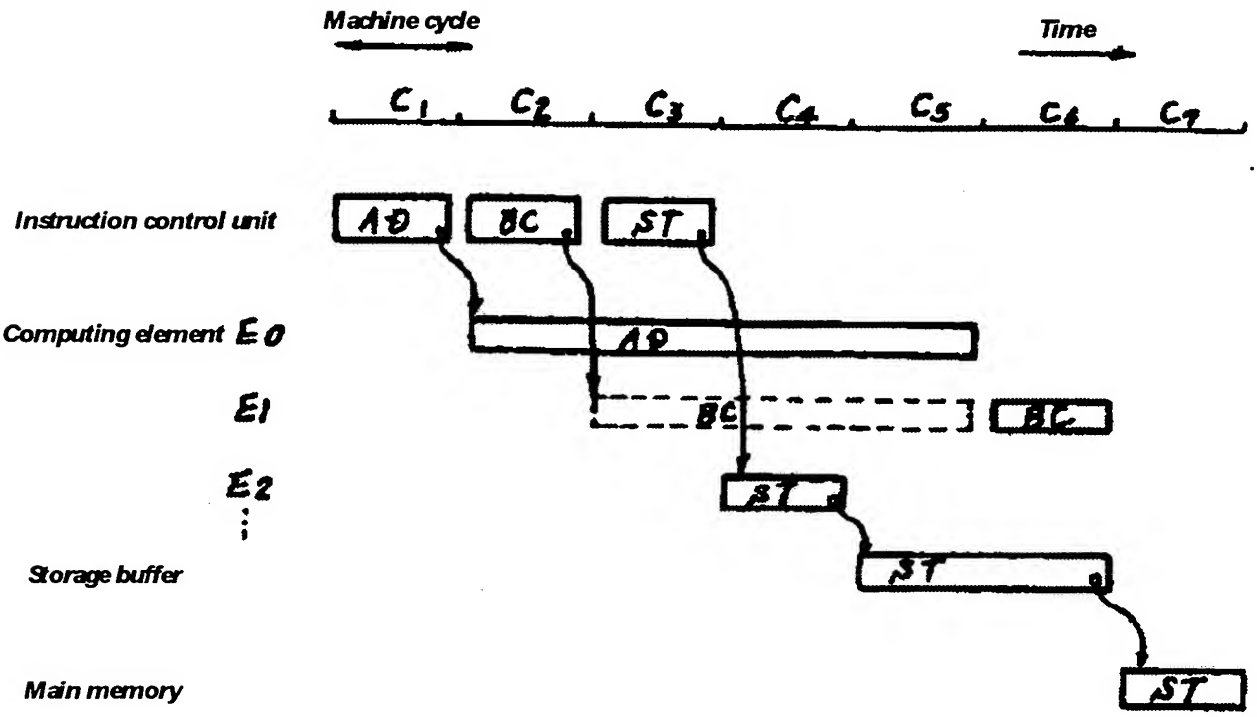

Machine cycle
Time
C1 C2 C3 C4 C5 C6 C7
Instruction control unit
AD
BC
ST
Computing element E0
AD
E1
BC
BC
E2
ST
ST
Storage buffer
Main memory
ST

FIG. 5

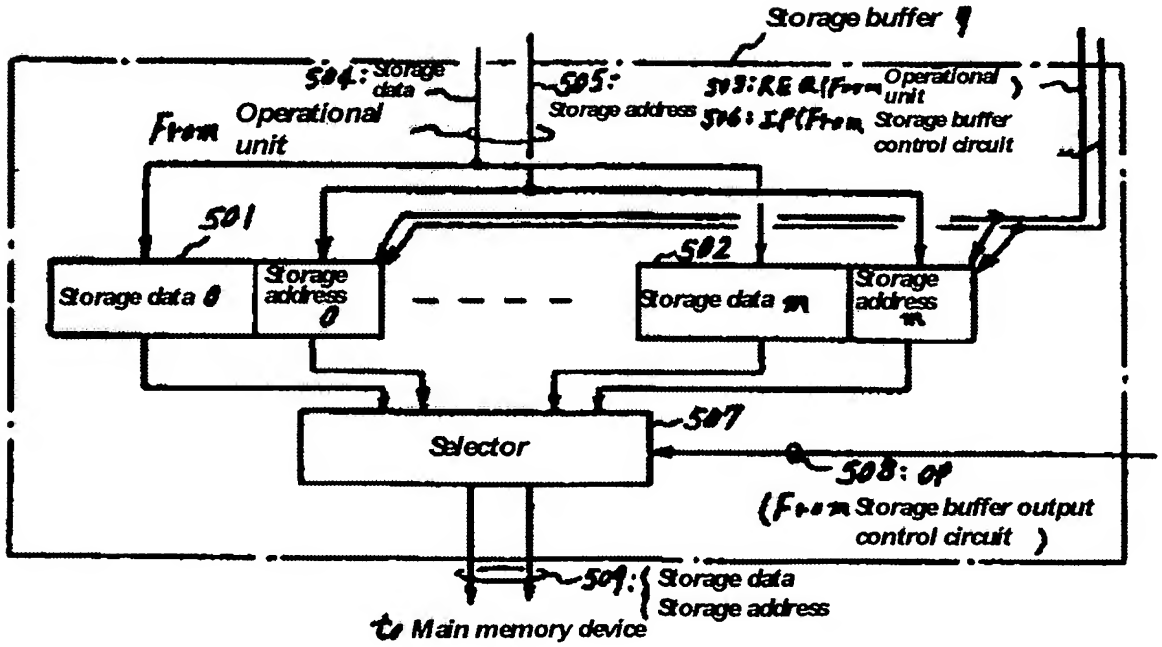
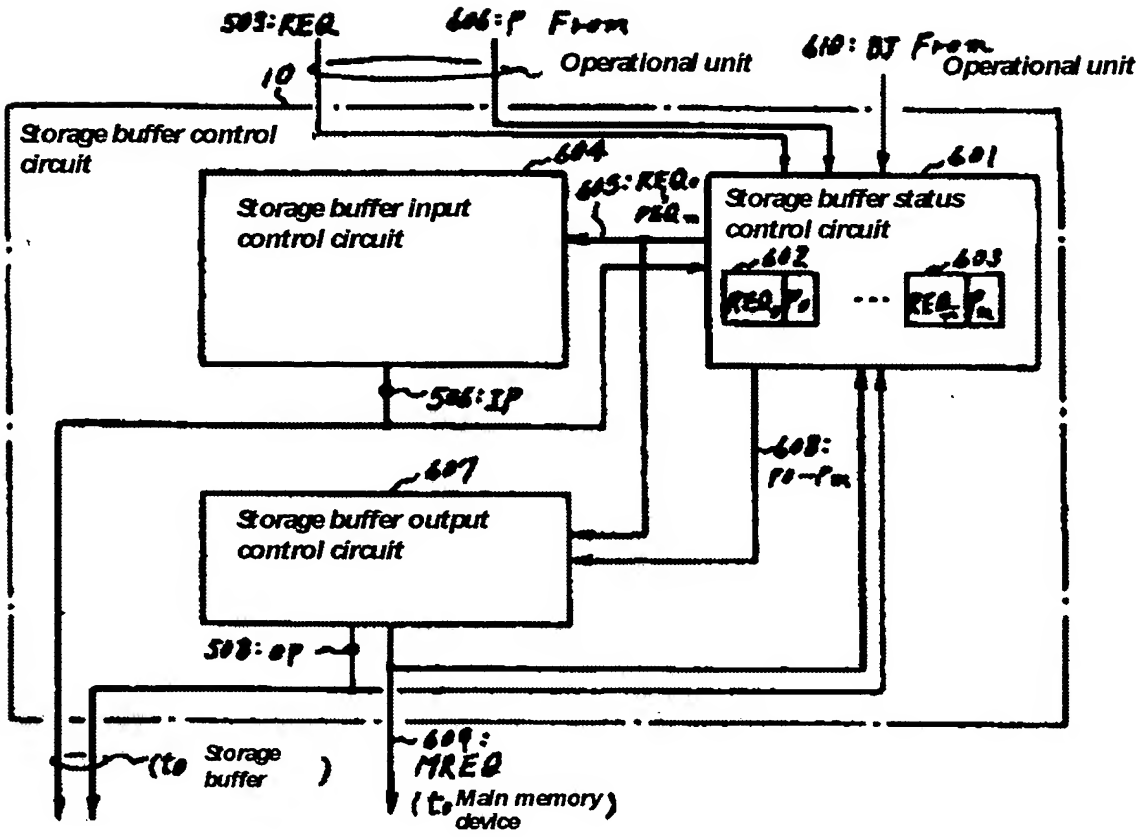

Storage buffer
504: Storage data
505: Storage address
From Operational unit
Storage buffer control circuit
501
502
Storage data 0
Storage address 0
Storage data m
Storage address m
Selector
507
508: OP
(From Storage buffer output control circuit)
509: Storage data
Storage address
to Main memory device

FIG. 6



Continuation from the 1st page:

(72) Inventor: Eiki KAMATA

Hitachi, Ltd., Center Research Institute

280, Higashikoigakubo 1-chome, Kokubunji-shi

(72) Inventor: Shigeo TAKEUCHI

Hitachi Microcomputer Engineering

1479, Josuihoncho, Kodaira-shi